UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/734,457	12/12/2003	Erik J. Johnson	42P16856	9691

8791 7590 10/03/2007
BLAKELY SOKOLOFF TAYLOR & ZAFMAN
1279 OAKMEAD PARKWAY
SUNNYVALE, CA 94085-4040

EXAMINER	
CHEN, QING	
ART UNIT	**PAPER NUMBER**
2191	
MAIL DATE	**DELIVERY MODE**
10/03/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

80

Office Action Summary	Application No.	Applicant(s)	
	10/734,457	JOHNSON ET AL.	
	Examiner	Art Unit	
	Qing Chen	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 25 July 2007.
2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.
3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-23 is/are pending in the application.
4a) Of the above claim(s) _____ is/are withdrawn from consideration.
5)☐ Claim(s) _____ is/are allowed.
6)☒ Claim(s) 1-23 is/are rejected.
7)☐ Claim(s) _____ is/are objected to.
8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.
10)☒ The drawing(s) filed on 09 July 2007 is/are: a)☐ accepted or b)☒ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
11)☒ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
a)☐ All b)☐ Some * c)☐ None of:
1.☐ Certified copies of the priority documents have been received.
2.☐ Certified copies of the priority documents have been received in Application No. _____.
3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date 20070709.	6) ☐ Other: _____.

DETAILED ACTION

1. This Office action is in response to the amendment filed on July 25, 2007.

2. **Claims 1-23** are pending.

3. **Claims 1-7 and 11-23** have been amended.

4. **Claims 24-29** have been cancelled.

5. The objection to the oath/declaration is maintained in view of Applicant's pending submission of the supplemental oath/declaration and further explained below.

6. The objection to drawings due to reference numbers not mentioned in the specification is withdrawn in view of Applicant's amendments to the drawings. However, the objection to the drawings due to missing shaded drawing elements is maintained in view of Applicant's amendments to the drawings and further explained below.

7. The objection to the title is withdrawn in view of Applicant's amendments to the title.

8. The objection to the abstract is withdrawn in view of Applicant's amendments to the abstract.

9. The objection to the specification is withdrawn in view of Applicant's amendments to the specification.

10. The objections to Claims 1-18 and 20-23 are withdrawn in view of Applicant's amendments to the claims. The objections to Claims 25 and 29 are withdrawn in view of Applicant's cancellation of the claims.

11. The 35 U.S.C. § 112, second paragraph, rejections of Claims 11-23 are withdrawn in view of Applicant's amendments to the claims. The 35 U.S.C. § 112, second paragraph, rejections of Claims 24-29 are withdrawn in view of Applicant's cancellation of the claims.

12. The 35 U.S.C. § 101 rejections of Claims 12-23 are withdrawn in view of Applicant's amendments to the claims. The 35 U.S.C. § 101 rejections of Claims 24-29 are withdrawn in view of Applicant's cancellation of the claims.

Response to Amendment

Oath/Declaration

13. The oath or declaration is defective. A new oath or declaration in compliance with 37 CFR 1.67(a) identifying this application by application number and filing date is required. See MPEP §§ 602.01 and 602.02.

> The oath or declaration is defective because:
> It does not state that the person making the oath or declaration believes the named inventor or inventors to be the first inventor or inventors of the subject matter which is claimed and for which a patent is sought.

The originally-filed oath/declaration lists plural named inventors and states that the person making the oath/declaration believes the named inventor is the **original and joint** inventor of the subject matter which is claimed and for which a patent is sought. However, 37 CFR 1.63(a)(4) recites that the oath/declaration must state that the person making the oath/declaration believes the named inventor or inventors to be the **original and first** inventor or inventors of the subject matter which is claimed and for which a patent is sought. The originally-filed oath/declaration only recites the "original and first" language for the scenario of one named inventor.

Drawings

14. The drawings are objected to because the originally-filed specification discloses that Figure 1A contains a shaded area *(see Page 3, Paragraph [0013])*. However, the replacement drawing sheet of Figure 1A appears to be identical to the originally-filed drawing sheet of Figure

1A. Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office action to avoid abandonment of the application.

Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. The figure or figure number of an amended drawing should not be labeled as "amended." If a drawing figure is to be canceled, the appropriate figure must be removed from the replacement sheet, and where necessary, the remaining figures must be renumbered and appropriate changes made to the brief description of the several views of the drawings for consistency. Additional replacement sheets may be necessary to show the renumbering of the remaining figures. Each drawing sheet submitted after the filing date of an application must be labeled in the top margin as either "Replacement Sheet" or "New Sheet" pursuant to 37 CFR 1.121(d). If the changes are not accepted by the Examiner, the Applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Claim Objections

15. **Claims 5, 7, and 16-18** are objected to because of the following informalities:

- **Claim 5** contains a typographical error: "the first second" should presumably read -- the first thread --.
- **Claims 7 and 18** contain a typographical error: "beginning a one of the nodes" should read -- beginning at one of the nodes --.

- **Claims 16 and 17** contain a typographical error: "wherein the instructions to cause to the first processor" should read -- wherein the instructions to cause the first processor --. Appropriate correction is required.

Claim Rejections - 35 USC § 112

16. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

17. **Claims 20 and 23** are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 20 recites the limitation "the processor." There is insufficient antecedent basis for this limitation in the claim. In the interest of compact prosecution, the Examiner subsequently interprets this limitation as reading "the second processor" for the purpose of further examination.

Claim 23 recites the limitation "the instructions." There is insufficient antecedent basis for this limitation in the claim. In the interest of compact prosecution, the Examiner subsequently interprets this limitation as reading "the stored instructions" for the purpose of further examination.

Claim Rejections - 35 USC § 102

18. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

19. **Claims 1-4, 8-16, and 19-22** are rejected under 35 U.S.C. 102(b) as being anticipated by <u>Song et al.</u> **(US 6,061,711)**.

As per **Claim 1**, <u>Song et al.</u> disclose:

- identifying instructions of a first thread that instruct a processor to perform a context swap to another thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.")*; and

- automatically inserting into the instructions of the first thread at least one additional instruction that instructs the processor to perform a context swap to another thread based on counts of consecutively executed instructions of the first thread that do not instruct the processor to perform a context swap to another thread *(see Figure 7; Column 8: 58-60, "... the marker is a conditional context switch instruction of co-processor 204 referred to as VCCS."; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an* *application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program." and 7-10, "... during the execution of the application program in program execution block 602, a thirty-two (32) bit VCCS instruction will be encountered at the appropriate point as shown in context switch instruction block 604.").*

As per **Claim 2**, the rejection of **Claim 1** is incorporated; and Song et al. further disclose:

- identifying instructions of a second thread that instruct a processor to perform a context swap to another thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.")*; and

- automatically inserting into instructions of a second thread at least one instruction that instructs the processor to perform a context swap to another thread based on counts of consecutively executed instructions of the second thread that do not instruct the processor to perform a context swap to another thread *(see Figure 7; Column 8: 58-60, "... the marker is a conditional context switch instruction of co-processor 204 referred to as VCCS."; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702* *are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program." and 7-10, "... during the execution of the application program in program execution block 602, a thirty-two (32) bit VCCS instruction will be encountered at the appropriate point as shown in context switch instruction block 604.")*;

- wherein processor is to swap execution among threads including the first thread and the second thread *(see Column 10: 57-61, "Referring to context switch request block 606, to determine whether or not processor 202 has requested that the currently executing application program in program execution block 602 be context switched out and replaced with another program ...").*

As per **Claim 3**, the rejection of **Claim 2** is incorporated; and Song et al. further disclose:

- automatically inserting into instructions of the first thread comprises inserting based on the identified instructions of the second thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.")*; and

- automatically inserting into instructions of the second thread comprises inserting based on the identified instructions of the first thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application* *program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.").*

As per **Claim 4**, the rejection of **Claim 2** is incorporated; and Song et al. further disclose:

- repeating a procedure that determines one or more locations to automatically insert instructions that instruct the processor to perform a context swap to another thread into the instructions of the first and second threads *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.").*

As per **Claim 8**, the rejection of **Claim 1** is incorporated; and Song et al. further disclose:

- wherein automatically inserting comprises inserting to keep intact a group of instructions identified as indivisible *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state* *information storage in anticipation of successfully and seamlessly context switching in the application program.").*

As per **Claim 9**, the rejection of **Claim 1** is incorporated; and Song et al. further disclose:

- wherein the processor comprises a multi-threaded central processor unit (CPU) *(see Column 3: 26-28, "A typical host processor 102 is an x86 processor such as an Intel Corporation Pentium™ or Pentium Pro™ processor.").*

As per **Claim 10**, the rejection of **Claim 1** is incorporated; and Song et al. further disclose:

- wherein the processor comprises a multi-threaded engine of a multi-engine processor *(see Column 3: 26-28, "A typical host processor 102 is an x86 processor such as an Intel Corporation Pentium™ or Pentium Pro™ processor.").*

As per **Claim 11**, the rejection of **Claim 10** is incorporated; and Song et al. further disclose:

- wherein the multi-threaded engine of the multi-engine processor comprises a multi-threaded engine not having any floating point instructions in the multi-threaded engine's instruction set *(see Column 3: 26-28, "A typical host processor 102 is an x86 processor such as an Intel Corporation Pentium™ or Pentium Pro™ processor.").*

As per **Claim 12**, Song et al. disclose:

- access instructions of a first thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.")*;

- identify instructions of the first thread that instruct a second processor to perform a context swap to another thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.")*; and

- insert into the instructions of the first thread at least one additional instruction that instructs the second processor to perform a context swap to another thread based on counts of consecutively executed instructions of the first thread that do not instruct the second processor to perform a context swap to another thread *(see Figure 7; Column 8: 58-60, "... the marker is a conditional context switch instruction of co-processor 204 referred to as VCCS."; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context* *switching in the application program." and 7-10, "... during the execution of the application program in program execution block 602, a thirty-two (32) bit VCCS instruction will be encountered at the appropriate point as shown in context switch instruction block 604.").*

As per **Claim 13**, the rejection of **Claim 12** is incorporated; and Song et al. further disclose:

- access instructions of a second thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.")*;

- identify instructions of the second thread that instruct the second processor to perform a context swap to another thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.")*; and

- insert into instructions of a second thread at least one instruction that instructs the second processor to perform a context swap to another thread based on counts of consecutively executed instructions of the second thread that do not instruct the processor to perform a context swap to another thread *(see Figure 7; Column 8: 58-60, "... the marker is a conditional context switch instruction of co-processor 204 referred to as VCCS."; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program." and 7-10, "... during the execution of the application program in program execution block 602, a thirty-two (32) bit VCCS instruction will be encountered at the appropriate point as shown in context switch instruction block 604.")*;

- wherein second processor is to swap execution among threads including the first thread and the second thread *(see Column 10: 57-61, "Referring to context switch request block 606, to determine whether or not processor 202 has requested that the currently executing application program in program execution block 602 be context switched out and replaced with another program ...").*

As per **Claim 14,** the rejection of **Claim 13** is incorporated; and Song et al. further disclose:

- insert into instructions of the first thread comprises inserting based on the identified instructions of the second thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state* *information storage in anticipation of successfully and seamlessly context switching in the application program.")*; and

- insert into instructions of the second thread comprises inserting based on the identified instructions of the first thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.").*

As per **Claim 15**, the rejection of **Claim 13** is incorporated; and Song et al. further disclose:

- repeat a procedure that determines one or more locations to automatically insert instructions that instruct the second processor to perform a context swap to another thread into the instructions of the first and second threads *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.").*

As per **Claim 16**, the rejection of **Claim 14** is incorporated; and Song et al. further disclose:

- wherein the instructions to cause the first processor to insert into instructions of the first thread based on the identified instructions comprise instructions to cause the first processor to insert into the instructions of the first thread based on a count of consecutive instructions that do not instruct the second processor to perform a context swap to another thread *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.").*

As per **Claim 19**, the rejection of **Claim 12** is incorporated; and Song et al. further disclose:

- wherein the instructions to cause the first processor to insert comprise instructions to insert to keep intact a group of instructions identified as indivisible *(see Figure 7; Column 9: 66 and 67 through Column 10: 1-5, "Referring to FIG. 7, an illustrative segment 700 of an application program includes interspersed VCCS instructions 702. The VCCS instructions 702 are generally regularly interspersed at predetermined locations requiring a minimal amount of processor state information storage in anticipation of successfully and seamlessly context switching in the application program.").*

As per **Claim 20**, the rejection of **Claim 12** is incorporated; and Song et al. further disclose:

- wherein the second processor comprises a multi-threaded central processor unit (CPU) *(see Column 3: 26-28, "A typical host processor 102 is an x86 processor such as an Intel Corporation Pentium™ or Pentium Pro™ processor.")*.

As per **Claim 21**, the rejection of **Claim 12** is incorporated; and Song et al. further disclose:

- wherein the second processor comprises a multi-threaded engine of a multi-engine processor *(see Column 3: 26-28, "A typical host processor 102 is an x86 processor such as an Intel Corporation Pentium™ or Pentium Pro™ processor.")*.

As per **Claim 22**, the rejection of **Claim 21** is incorporated; and Song et al. further disclose:

- wherein the multi-threaded engine of the multi-engine processor comprises a multi-threaded engine not having any floating point instructions in the multi-threaded engine's instruction set *(see Column 3: 26-28, "A typical host processor 102 is an x86 processor such as an Intel Corporation Pentium™ or Pentium Pro™ processor.")*.

Claim Rejections - 35 USC § 103

20. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

21. **Claims 5, 6, and 17** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Song et al.** **(US 6,061,711)** in view of **Chrysos et al.** **(US 5,809,450)**.

As per **Claim 5**, the rejection of **Claim 3** is incorporated; however, Song et al. do not disclose:

- wherein the automatically inserting into instructions of the first thread based on the identified instructions comprises inserting into the instructions of the first thread based on an average number of consecutive instructions of the first thread that do not instruct the processor to perform a context swap to another thread.

Chrysos et al. disclose:

- wherein the automatically inserting into instructions of the first thread based on the identified instructions comprises inserting into the instructions of the first thread based on an average number of consecutive instructions of the first thread that do not instruct the processor to perform a context swap to another thread *(see Column 18: 19-22, "... function 755 takes as input state information 756 such as addresses, process identifiers, address space numbers, hardware context identifiers, or thread identifiers of the selected instructions." and 30-34, "Step 760* *produces a subset of samples based on the function 755. In step 780, statistics 790 are determined. These statistics can include averages, standard deviations, histograms (distribution), and error bounds of the properties of the sampled instructions.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Chrysos et al. into the teaching of Song et al. to include wherein the automatically inserting into instructions of the first thread based on the identified instructions comprises inserting into the instructions of the first thread based on an average number of consecutive instructions of the first thread that do not instruct the processor to perform a context swap to another thread. The modification would be obvious because one of ordinary skill in the art would be motivated to show the distribution of instruction execution, memory access rates, or latencies *(see Chrysos et al. – Column 18: 38-39).*

As per **Claim 6**, the rejection of **Claim 3** is incorporated; however, Song et al. do not disclose:

- wherein the automatically inserting into instructions of the first thread based on the identified instructions comprises inserting into the instructions of the first thread base on a standard deviation derived from a number of consecutive instructions of the second thread that do not instruct the processor to perform a context swap to another thread.

Chrysos et al. disclose:

- wherein the automatically inserting into instructions of the first thread based on the identified instructions comprises inserting into the instructions of the first thread base on a standard deviation derived from a number of consecutive instructions of the second thread that do not instruct the processor to perform a context swap to another thread *(see Column 18: 19-22, "... function 755 takes as input state information 756 such as addresses, process identifiers, address space numbers, hardware context identifiers, or thread identifiers of the selected instructions." and 30-34, "Step 760 produces a subset of samples based on the function 755. In step 780, statistics 790 are determined. These statistics can include averages, standard deviations, histograms (distribution), and error bounds of the properties of the sampled instructions.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Chrysos et al. into the teaching of Song et al. to include wherein the automatically inserting into instructions of the first thread based on the identified instructions comprises inserting into the instructions of the first thread base on a standard deviation derived from a number of consecutive instructions of the second thread that do not instruct the processor to perform a context swap to another thread. The modification would be obvious because one of ordinary skill in the art would be motivated to show the distribution of instruction execution, memory access rates, or latencies *(see Chrysos et al. – Column 18: 38-39).*

As per **Claim 17**, the rejection of **Claim 16** is incorporated; however, Song et al. do not disclose:

- wherein the instructions to cause the first processor to insert into instructions of the first thread based on the identified instructions comprise instructions to cause the first processor to insert into the instructions of the first thread based on a standard deviation derived from the number of consecutive instructions that do not instruct the second processor to perform a context swap to another thread.

Chrysos et al. disclose:

- wherein the instructions to cause the first processor to insert into instructions of the first thread based on the identified instructions comprise instructions to cause the first processor to insert into the instructions of the first thread based on a standard deviation derived from the number of consecutive instructions that do not instruct the second processor to perform a context swap to another thread *(see Column 18: 19-22, "... function 755 takes as input state information 756 such as addresses, process identifiers, address space numbers, hardware context identifiers, or thread identifiers of the selected instructions." and 30-34, "Step 760 produces a subset of samples based on the function 755. In step 780, statistics 790 are determined. These statistics can include averages, standard deviations, histograms (distribution), and error bounds of the properties of the sampled instructions.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Chrysos et al. into the teaching of Song et al. to include wherein the instructions to cause the first processor to insert into instructions of the first thread based on the identified instructions comprise instructions to cause the first processor to insert into the instructions of the first thread based on a standard deviation derived from the number of consecutive instructions that do not instruct the second processor to perform a context swap to another thread. The modification would be obvious because one of ordinary skill in the art would be motivated to show the distribution of instruction execution, memory access rates, or latencies *(see Chrysos et al. – Column 18: 38-39).*

22. **Claims 7, 18, and 23** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Song et al. (US 6,061,711)** in view of **Dubey et al. (US 5,812,811)**.

As per **Claim 7**, the rejection of **Claim 1** is incorporated; however, Song et al. do not disclose:

- constructing a data flow graph of the instructions of the first thread, the data flow graph comprising an organization of nodes associated with subsets of the instructions of the first thread; and

- determining at least one of the following: a number of consecutive instructions ending at one of the nodes that do not instruct the processor to perform a context swap to another thread; a number of consecutive instructions beginning at one of the nodes that do not instruct the processor to perform a context swap to another thread; and a number of consecutive instructions between instructions of one of the nodes that instruct the processor to perform a context swap to another thread.

Dubey et al. disclose:

- constructing a data flow graph of the instructions of the first thread, the data flow graph comprising an organization of nodes associated with subsets of the instructions of the first thread *(see Figures 5A, 5B, and 6; Column 21: 14-16, "Code sequences shown have been broken into blocks of non-branch instructions, optionally ending with a branch instruction." and 29-30, "... different control independent blocks, such as, B1 and B12, in FIG. 5.")*; and

- determining at least one of the following: a number of consecutive instructions ending at one of the nodes that do not instruct the processor to perform a context swap to another thread; a number of consecutive instructions beginning at one of the nodes that do not instruct the processor to perform a context swap to another thread; and a number of consecutive instructions between instructions of one of the nodes that instruct the processor to perform a context swap to another thread *(see Column 21: 45-50, "Conditional suspend, or, SUSPEND instruction is used in block B9 to speculatively execute next two instructions, assuming the forking thread (executing block B3) flows into block B7 at run time and avoids block B6 (which updates register 3), as a result of the branch at the end of block B3." and 52 and 53, "... SUSPEND instruction is used to speculatively execute next four instructions.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Dubey et al. into the teaching of Song et al. to include constructing a data flow graph of the instructions of the first thread, the data flow graph comprising an organization of nodes associated with subsets of the instructions of the first thread; and determining at least one of the following: a number of consecutive instructions ending at one of the nodes that do not instruct the processor to perform a context swap to another thread; a number of consecutive instructions beginning at one of the nodes that do not instruct the processor to perform a context swap to another thread; and a number of consecutive instructions between instructions of one of the nodes that instruct the processor to perform a context swap to another thread. The modification would be obvious because one of ordinary skill in the art would be motivated to exploit parallelism at run time *(see Dubey et al. – Column 2: 41-47).*

As per **Claim 18**, the rejection of **Claim 12** is incorporated; however, Song et al. do not disclose:

- construct a data flow graph of the instructions of the first thread, the data flow graph comprising an organization of nodes associated with subsets of the instructions of the first thread; and

- determine at least one of the following: a number of consecutive instructions ending at one of the nodes that do not instruct the second processor to perform a context swap to another thread; a number of consecutive instructions beginning at one of the nodes that do not instruct the second processor to perform a context swap to another thread; and a number of consecutive instructions between instructions of one of the nodes that instruct the second processor to perform a context swap to another thread.

Dubey et al. disclose:

- construct a data flow graph of the instructions of the first thread, the data flow graph comprising an organization of nodes associated with subsets of the instructions of the first thread *(see Figures 5A, 5B, and 6; Column 21: 14-16, "Code sequences shown have been broken into blocks of non-branch instructions, optionally ending with a branch instruction." and 29-30, "... different control independent blocks, such as, B1 and B12, in FIG. 5.")*; and

- determine at least one of the following: a number of consecutive instructions ending at one of the nodes that do not instruct the second processor to perform a context swap to another thread; a number of consecutive instructions beginning at one of the nodes that do not instruct the second processor to perform a context swap to another thread; and a number of consecutive instructions between instructions of one of the nodes that instruct the second processor to perform a context swap to another thread *(see Column 21: 45-50, "Conditional suspend, or, SUSPEND instruction is used in block B9 to speculatively execute next two instructions, assuming the forking thread (executing block B3) flows into block B7 at run time and avoids block B6 (which updates register 3), as a result of the branch at the end of block B3." and 52 and 53, "... SUSPEND instruction is used to speculatively execute next four instructions.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Dubey et al. into the teaching of Song et al. to include construct a data flow graph of the instructions of the first thread, the data flow graph comprising an organization of nodes associated with subsets of the instructions of the first thread; and determine at least one of the following: a number of consecutive instructions ending at one of the nodes that do not instruct the second processor to perform a context swap to another thread; a number of consecutive instructions beginning at one of the nodes that do not instruct the second processor to perform a context swap to another thread; and a number of consecutive instructions between instructions of one of the nodes that instruct the second processor to perform a context swap to another thread. The modification would be obvious because one of ordinary skill in the art would be motivated to exploit parallelism at run time *(see Dubey et al. – Column 2: 41-47)*.

As per **Claim 23**, the rejection of **Claim 22** is incorporated; however, Song et al. do not disclose:

- wherein the stored instructions comprise instructions of at least one of the following: a compiler, an assembler, and a source code pre-processor.

Dubey et al. disclose:

- wherein the stored instructions comprise instructions of at least one of the following: a compiler, an assembler, and a source code pre-processor *(see Column 4: 54-58, "The invention discloses novel Fork-Suspend instructions that are added to the instruction set of the CPU, and are inserted in a program prior to run-time to delineate potential future threads for parallel execution. Preferably, this is done by a compiler.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Dubey et al. into the teaching of Song et al. to include wherein the stored instructions comprise instructions of at least one of the following: a compiler, an assembler, and a source code pre-processor. The modification would be obvious because one of ordinary skill in the art would be motivated to group together control independent and data-independent instructions *(see Dubey et al. – Column 2: 17-19)*.

Response to Arguments

23. Applicant's arguments with respect to Claims 1 and 12 have been considered, but are moot in view of the new ground(s) of rejection.

Conclusion

24. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL.** See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

The prior art made of record and not relied upon is considered pertinent to Applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Qing Chen whose telephone number is 571-270-1071. The Examiner can normally be reached on Monday through Thursday from 7:30 AM to 4:00 PM. The Examiner can also be reached on alternate Fridays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Wei Zhen, can be reached on 571-272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]

WEI ZHEN
SUPERVISORY PATENT EXAMINER

QC / [illegible]
September 20, 2007